**Corso di Programmazione Web e Mobile**

**A.A. 2017-2018**

**Planet Defender Game**

Giovanni Sorgente [911248]

# Planet Defender

Una game app basata sul concetto di RTS (Real Time Strategy)

# Introduzione

Planet Defender, vuole essere una game app, basata su Angular 6 che si prefigge l’obiettivo di infoltire il parco applicazioni ludico fruibile sia da browser che da smartphone (Android).

La direttiva che sta alla base del progetto è quella di creare una sessione di gioco in modalità arcade, che permetta all’utente, in maniera del tutto gratuita di distrarsi e riempire quei “tempi morti” che spesso capitano durante la giornata.

Ad oggi, troviamo nei vari store sempre più applicazioni che presentano contenuti solitamente ridotti ma dall’obiettivo mirato al “divertimento” e svago temporaneo dell’utente.

Prendiamo ad esempio i casi più eclatanti che possono tornare alla mente, come Candy Crush oppure il meno famoso Flappy Bird; la richiesta è quella di avere un accesso veloce alla modalità di gioco, eventuali perks dovuti alla partita interrompibile in qualsiasi momento ed uno stretto ciclo di vita della partita che può essere ripetuta infinitamente rilassando l’utente che in quel frangente non dovrà impegnare troppo intensamente le proprie capacità intellettive ma potrà comunque trovare soddisfazione dall’attività svolta risolvendo i piccoli e semplici enigmi posti in essere dal gioco stesso.

A mio avviso la vastità del parco software presente online oggi permette sicuramente di usufruire di qualsiasi tipologia di applicazione si stia cercando; ovvero credo sia molto complicato trovare idee innovative che siano davvero originali e che possano essere poi proposte al grande pubblico.

Sono piuttosto convinto che la continua ricerca da parte dell’essere umano di qualcosa di diverso, di nuovo, di mai visto; porti poi, anche in questo ambito, al medesimo risultato del non rendere mai del tutto coperta una determinata tipologia di applicazioni.

Ecco che una piccola variante può così dal nulla creare una nuova versione che stimola di nuova la voglia di spendere del tempo da parte dell’utente nel superare quel determinato livello o sconfiggere quel determinato avversario.

## Breve analisi dei requisiti

### Destinatari

**Capacità e possibilità tecniche**.

Quale livello di esperienza possiedono?

Conoscono già cosa cercano o devono essere guidati?

Quale quantità di banda disponibile?

Che tipo di device useranno per connettersi?

**Linguaggio.**

Di quali linguaggi hanno esperienza?

Ci sono per loro dei linguaggi empatici?

Ci sono dei linguaggi o degli stili di riferimento per il tipo di applicazione proposta?

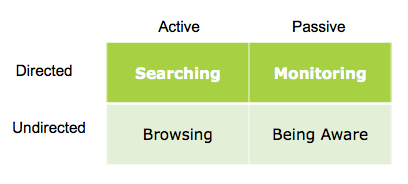
**Motivazione.**

Tipo di motivazione: intrattenimento, business, educational.

Livello di consapevolezza: predisposizione alla ricerca diretta o indiretta delle informazioni

Livello di motivazione: predisposizione alla ricerca attiva o passiva

Utilizzando il modello di Bates definire quindi le strategie adeguate per l'organizzazione dei contenuti.



### Modello di valore

Il valore dell'applicazione.

Quali servizi o contenuti danno valore all'applicazione?

Quali elementi possono attirare l'interesse degli utenti?

Quali elementi forniscono un valore all'investitore?

È possibile identificare le transazioni che generano questo valore?

È possibile fare una stima del valore prodotto (denaro, attenzione utenti, altro)?

### Flusso dei dati

Quale qualità, stile e livello di dettagli caratterizzano i contenuti?

**Ottenere i contenuti**.

I contenuti vanno prodotti o possono essere reperiti sul mercato?

Quali sono i costi di produzione o riadattamento?

**Archiviare e organizzare i contenuti**

Quale metodo di archiviazione?

Quali metodi di selezione dei contenuti?

**Pubblicare i contenuti**

Quali trasformazioni devono essere applicate?

I contenuti devono essere resi in formati che facilitino la condivisione (riuso) o questa deve essere sfavorita?

Con che frequenza devo aggiornare i contenuti?

Come scelgo un contenuto?

Ci sono vincoli alla pubblicazione (diritti, licenze, scadenza)?

### Aspetti tecnologici

Sandard per i contenuti. Come gestisco la portabilità dei contenuti in archiviazione? Che formati usare? Quali sono gli standard? Come gestisco la portabilità dei contenuti in pubblicazione? Come li seleziono? Come li compongo?

Gestione del Codice tramite modello MVC (modelli alternativi da concordare col docente).

Tecnologie utilizzate. Come sono state utilizzate le tecnologie (elencarle distinguendo quelle richieste dal progetto d'esame e quelle eventualmente aggiunte per altri motivi)? Ci sono motivazioni che hanno portato a scegliere una soluzione scartandone altre?

Javascript

Linguaggio di scripting creato da Brendan Eich nel 1995; a mio parere uno dei più flessibili e comodi linguaggi presenti ad oggi per la creazione di applicazioni di vario genere.

Pur essendo nato principalmente ai tempi con l’obiettivo di costruire script dinamici nei primi browser nascenti, oggi si afferma sempre più come linguaggio utilizzato anche per lo sviluppo di servizi con il noto NodeJs.

Typescript

Definito in genere come un “super set” di istruzioni costruito sopra Javascript,

Snippets e modalità di utilizzo

<creazione api, modello codice backend, … >

***Riferimenti***

Angular 6/Angular-cli

Angular ad oggi in versione 6, nasce nel 2010 con una prima versione scritta interamente in javascript che si basava fortemente sul pattern MVC.

L’idea alla base del framework è quella di poter definire modelli dati, viste e controllori ben disaccoppiati tra loro.

…..AAAAAAAAAA...

Ma qual’è inoltre il grosso vantaggio che un’azienda dovrebbe considerare nel seguire

un approccio di questo tipo?

Il principale vantaggio da evidenziare nell’adozione di questo modello di sviluppo, sta sicuramente nel fatto che un team di sviluppo è in grado di separare e fornire più agilmente il lavoro ai componenti del gruppo, dato che il framework stesso nasce con l’idea di garantire questa separazione di ambiti.

Garantisce sicuramente uno standard nella modalità di lavoro, questo permette certamente l’inserimento di una nuova risorsa in tempi più brevi.

Snippets e modalità di utilizzo

<creazione api, modello codice backend, … >

***Riferimenti***

[***https://angular.io/***](https://angular.io/)

MongoDB

La scelta di MongoDB, ad oggi uno tra i più conosciuti DBMS non relazionali, nasce dall’esigenza di poter gestire una grossa mole di dati senza uno schema database ben preciso.

Basti pensare ad esempio alla necessità di salvare le statistiche utente che, a fronte di un grosso numero di registrazioni, potrebbero richiedere il salvataggio di un gran numero di statistiche ed informazioni da salvare.

Un futuro sviluppo dell’applicazione potrebbe ad esempio dare la possibilità di visionare un match in corso, fornendo agli spettatori funzionalità di commentare la partita o marchiare con una spunta di gradimento le mosse dei giocatori.

Tutto questo, visto in larga scala, potrebbe sicuramente essere reso più performante denormalizzando le strutture dati in essere e gestendo il tutto senza relazioni tra gli oggetti.

Snippets e modalità di utilizzo

<creazione api, modello codice backend, … >

***Riferimenti***

Ionic

….

Snippets e modalità di utilizzo

<creazione api, modello codice backend, … >

***Riferimenti***

Node.js/Javascript

Nodejs ….

Snippets e modalità di utilizzo

<creazione api, modello codice backend, … >

***Riferimenti***

*https://nodejs.org/it/*

Swagger/Open Api 3.0

…

***Riferimenti***

*https://nodejs.org/it/*

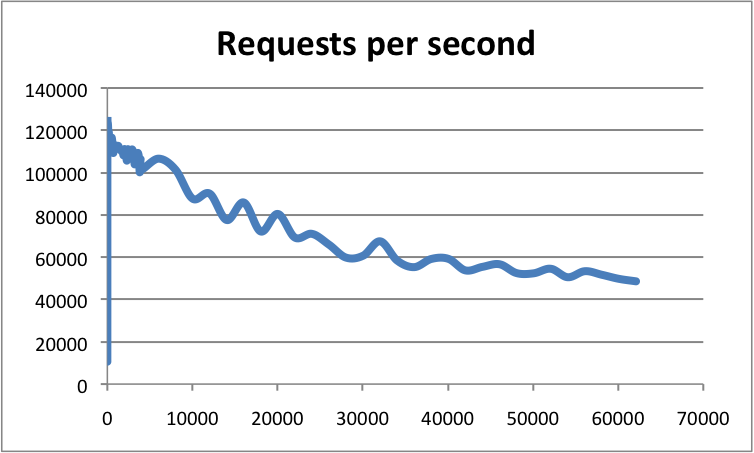
Redis

Redis è fa parte della categoria di prodotti cosiddetti datastore in memory. Sviluppato inizialmente da un italiano, Salvatore Sanfilippo che decide di mantenere il progetto open source (<https://github.com/antirez/redis>), diventa nel breve periodo una community ben più ampia e che oggi probabilmente deve la propria popolarità al gruppo Redis Labs che ha pensato alla commercializzazione di servizi per distribuire il prodotto stesso.

L’adozione di Redis nasce dall’esigenza di persistere in maniera volatile dei dati di gioco in maniera rapida e veloce.

Serve inoltre un meccanismo di publish e subscribe, in grado di notificare/venire a conoscenza delle mosse dei due giocatori per poter poi informare il corrispettivo avversario dell’azione effettuata.

A questo punto, si è scelto di proseguire con la tecnologia in oggetto data la comprovata rapidità nel gestire le richieste, documentata anche dai benchmarks forniti direttamente sul sito del prodotto (<https://redis.io/topics/benchmarks>).



Snippets e modalità di utilizzo

<code snippet>

***Riferimenti***

[*https://github.com/noderedis/node\_redis*](https://github.com/noderedis/node_redis)

*https://redis.io/*

### Altro

# Interfacce

Descrivere le principali interfacce utente dell'applicazione. È importante evidenziare quale stile sarà applicato alle interfacce e quali risorse saranno coinvolte nella loro esecuzione.

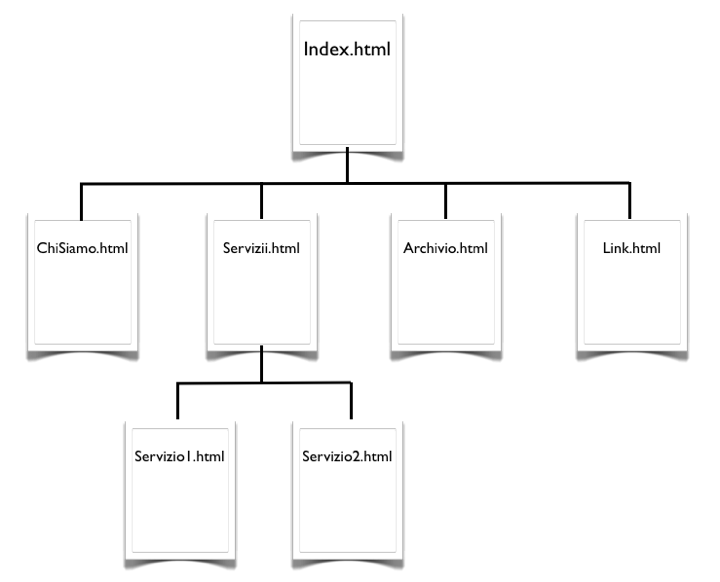
Le interfacce possono essere disegnate o tramite un software o semplicemente a mano.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Architettura

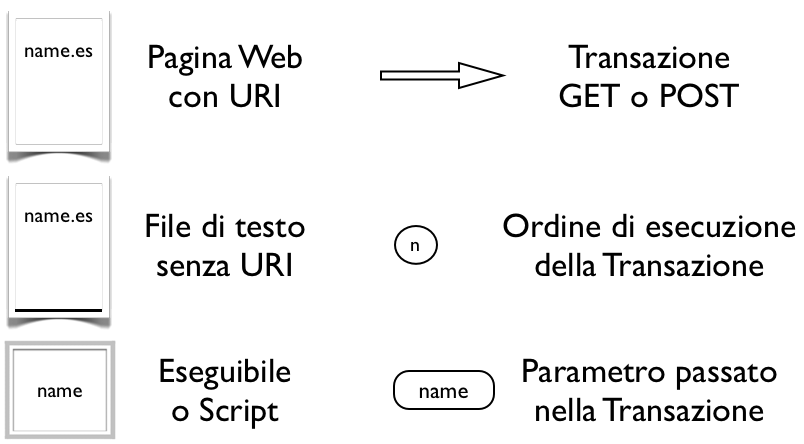
## Diagramma dell'ordine gerarchico delle risorse

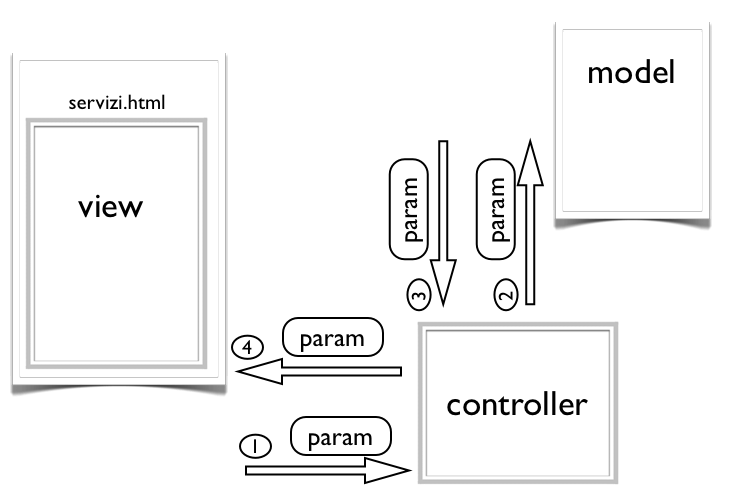
Identificazioni delle pagine da rendere accessibili come risorse (URI) e loro organizzazione gerarchica.

Figura 1: Architettura del sito

## Descrizione delle risorse

Ogni singola risorsa deve essere descritta per le sue componenti MVC indicando le transazioni, l'ordine delle transazioni e gli eventuali parametri passati nella transazione.

Figura 2: Legenda della notazione utilizzata

Figura 3: Componenti MVC della pagina servizi.html

## Altri diagrammi

*Eventuali altri diagrammi, come ad esempio il diagramma del DB*

# Codice

*Frammenti del codice più significativo*

## HTML

## CSS

## API

## Node.js

# Conclusioni

## Heading 1

text

## Heading 2

|  | column 2 | column 3 | column 4 |
| --- | --- | --- | --- |
| item 1 | text text text | text text text text text | R 12 581.36 |
| item 2 | text text text text text | text text text text text | R 5 149.35 |
|  |  |  | R 25.60 |
|  |  | Total due: | \*\* Expression is faulty \*\* |

# Nota bibliografica e sitografica

*Dove si elencano le risorse bibliografiche e sitografiche usate come riferimento e documentazione del lavoro.*

1. Autore, Titolo, Editore, Anno
2. URL, (autore data), Data di consultazione, (URN)

Modalità di esame nella discussione del progetto

Progetto

Il progetto consiste nella realizzazione di un'applicazione Web o mobile.

Per prima cosa si richiede di definire i requisiti fondamentali che l'applicazione soddisferà. A questo punto l'obiettivo è progettare, attraverso le tecnologie presentate a lezione, un'applicazione che realizzi le scelte tecnologiche coerentemente con i requisiti applicativi.

Per la valutazione della prova orale vengono tenuti in considerazione i seguenti aspetti:

1. capacità di usare tutte le tecnologie presentate a lezione;
2. complessità del progetto sviluppato;
3. qualità della progettazione e della relazione che la documenta;
4. esposizione: capacità di presentare il progetto e di motivare le scelte tecnologiche.

Sono ammessi gruppi composti al massimo da 2 studenti.

L'applicazione deve usare in maniera integrata tutte le tecnologie discusse.

L'applicazione deve essere il risultato originale del lavoro esclusivo dei componenti del gruppo di progetto.

Tutti i componenti del gruppo devono conoscere ogni dettaglio del progetto presentato.

Durante l'orale verranno discussi tutti gli aspetti del progetto: organizzazione, uso delle tecnologie, realizzazione (cioè verrà esaminato il codice).

Il tema di applicazione del progetto potrà essere proposto dagli studenti oppure scelto tra le proposte del docente (pubblicate nell'apposita sezione).

**Requisiti indispensabili del progetto**

Il progetto deve presentare le seguenti caratteristiche:

HTML5/CSS

Le pagine devono essere sviluppate in formato HTML5.

Tutte le pagine devono essere validate.

Il layout delle pagine deve essere sviluppato con CSS.

L'applicazione dovrà servirsi di almeno una API HTML5.

AJAX

Il progetto deve implementare una o più chiamate XMLHttpRequest.

Le chiamate possono interrogare dati in JSON, XML, XHTML, TXT.

NodeJS

Il progetto deve implementare una o più chiamate a un servizio NodeJS.

*Le chiamate devono richiedere o caricare dati in JSON o XML.*